haute école neuchâtel berne jura arc ingénierie www.he-arc.ch

Rapport

Titre	Laboratoire 1.1 : *Pattern Bridge*
Professeur	Stéphane Gobron
Étudiants	Sébastien Vaucher, Diego Antognini, Alexandre Perez
Date	28 mars 2013

Le *Pattern*

Le but premier du *pattern Bridge* est de créer une abstraction totalement indépendante de l'implémentation concrète. De ce fait, les classes entités (dans notre cas : les formes) ne sont plus dépendantes de la manière dont l'implémentation matérielle est réalisée.

La grande force du *pattern* est de permettre à notre programme d'être compatible avec plusieurs API sans en être dépendant. Il en découle une certaine facilité d'adaptation.

Notre projet

Le but du projet est de pouvoir dessiner différents types de formes sur plusieurs modèles d'implémentations matérielles. Le rendu visuel de la forme se comportera de la même manière, quelle que soit l'interface de dessin utilisée.

Pour l'implémentation sans le *pattern*, nous avons réalisé une solution intuitive au problème : nous utilisons une classe abstraite *Shape*, représentant l'ensemble des formes. Pour chaque type de forme, nous créons autant de classes spécialisées qu'il y a d'implémentations matérielles.

En ce qui concerne la réalisation avec le *pattern Bridge*, nous avons respecté la structure de classes imposée par la consigne.

Comparaison des deux solutions

Admettons que l'on aimerait ajouter un triangle à nos formes. Dans le cas sans *Bridge*, nous devrions créer une classe *Triangle* puis autant de classes fils *TriangleVx* qu'il y a d'implémentations matérielles.

Avec *Bridge*, il suffit de créer une et une seule classe *Triangle*, spécialisant la classe *Shape*.

Nous aimerions rendre notre programme compatible avec une nouvelle implémentation matérielle. Sans *Bridge*, il nous faudrait ajouter une nouvelle classe concrète utilisant la nouvelle mise en œuvre pour chaque forme.

En utilisant *Bridge*, il faudrait tout simplement créer une classe enfant héritant de la classe *Drawing*. Cette dernière s'occupe d'adapter nos méthodes de dessin à la nouvelle interface.

Conclusion

Dans le cas où aucun *pattern* n'est utilisé, il apparaît que l'ajout de nouvelles formes ou implémentations matérielles devient chronophage et la maintenance de plus en plus complexe.

Grâce à *Bridge*, ces opérations nécessitent au plus la création d'une seule nouvelle classe.